The Raspberry Pi Alternative for *WSJT-X*

Turn an inexpensive microcomputer into a digital communication machine.

Thomas Kocourek, N4FWD

In the "Eclectic Technology" column in the April 2017 issue of QST, *Chuck Kelly, W9MDO/VE1MDO, described a portable option for running* WSJT-X *digital mode software with a Raspberry Pi 3 microcomputer and a miniature LCD touchscreen. In this article, Thomas Kocourek, N4FWD, presents a similar solution, but with the emphasis on using the Pi as a dedicated home station computer for JT65, JT9, and WSPR — Ed.*

The author's station, with his Raspberry Pi 3 microcomputer at lower left, in its transparent case.

As we slide into the oncoming solar minimum, it's no surprise that we're seeing an uptick in popularity for digital modes, such as JT65 and JT9, as well as the WSPR digital beacon mode. JT65 and JT9 can support contacts on the HF bands under conditions that would render other modes unusable. And for those interested in HF propagation studies, WSPR is ideal.

All three modes are available in the free *WSJT-X* software package, created by Dr. Joe Taylor, K1JT. In addition to an SSB transceiver, all you need is a computer to run the software and a device to interface the computer to your radio, unless your radio has an interface built in.

While any desktop or laptop computer can do the job (*WSJT-X* is available for *Windows*, *MacOS*, and *Linux*), many amateurs have discovered an alternative that's inexpensive and fun to use: the Raspberry Pi 3.

Gathering the Hardware

The first step on the path to your Raspberry Pi station for *WSJT-X* is to gather the necessary hardware. You may already have bits and pieces around your ham shack. However, here is a list for those starting from scratch:

- **A Raspberry Pi 3 microcomputer**. For beginners, I strongly recommend a "kit," such as those offered by CanaKit (see Amazon and other sources), because these packages include almost everything you'll need, often including a case and power supply. Prices range from about $50 to $90. The power supply is critical to the proper operation of the Pi. It must be well filtered and provide 5 V dc at 2.4 A. The supply must also have a micro-USB connector to mate with the power port on the computer.
- **A monitor with an HDMI input**. The Raspberry Pi will work with most monitor resolutions, but a higher resolution monitor is much easier on the eyes. If your monitor supports DVI-D rather than HDMI, you can use a DVD-D-to-HDMI adapter. They're available from many sources.
- **An HDMI cable**. This is the cable that will link your Raspberry Pi to your monitor. Considering the small size of the Raspberry Pi, I'd recommend a lightweight cable to keep everything mechanically stable.
- **A USB "A-B" style cable**. This cable will link your Raspberry Pi to your interface or transceiver.
- **A keyboard and mouse**. To keep cabling to a minimum, I recommend a wireless keyboard and mouse. The Raspberry Pi has Bluetooth wireless capability built in, and the setup is easy.
- **Interface**. See Figure 1. Assuming your transceiver doesn't have a digital-mode interface of its own, you'll need to purchase an external interface, such as one made by MFJ, Tigertronics, West Mountain Radio, and others. Interfaces have various features; some are strictly for managing transmit/receive switching and audio, while others allow the Pi to "talk" to the transceiver and exchange information, such as the operating frequency.

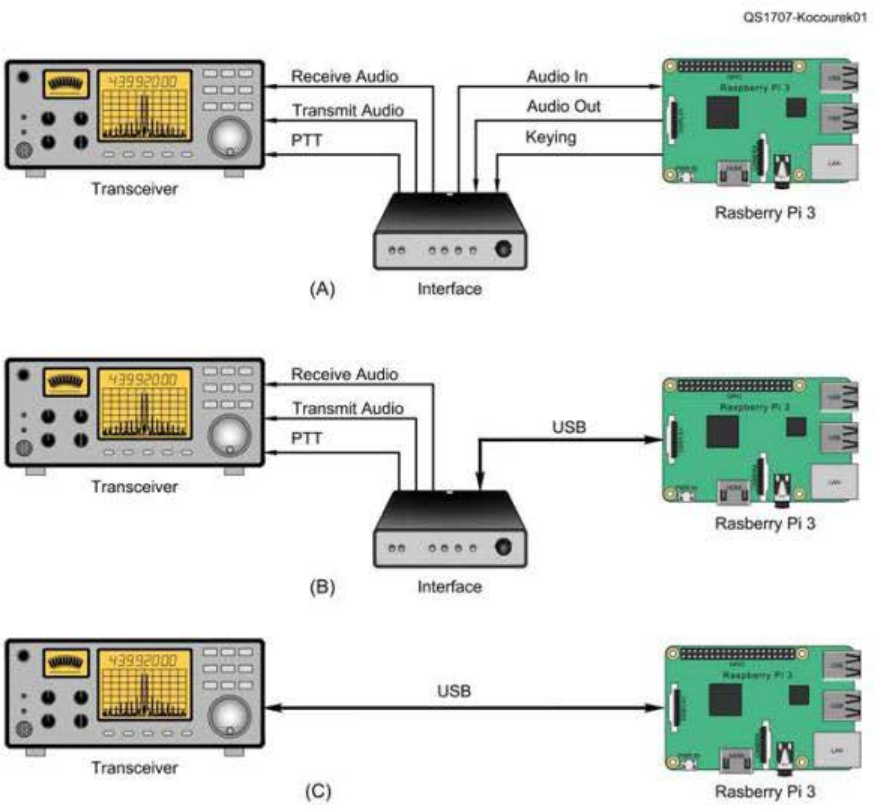


Figure 1 — Three of the most common configurations for interfacing your transceiver with your Raspberry Pi 3. (A) The most common setup for many years was based on an interface that took transmit and receive audio from the computer sound card, along with transmit/receive keying from one of the computer's COM ports, and passed everything to the transceiver in a way that kept the signal lines isolated. This is increasingly less common today, but if you wish to do this with a Raspberry Pi 3, you'll need to use an inexpensive USB sound device plugged into one of the Raspberry Pi 3 USB ports. (B) This is becoming the most common arrangement, because many modern interfaces now have built-in sound devices and USB connectivity. (C) The future will likely see the disappearance of the interface entirely as transceivers incorporate their own computer interfacing. Today, a number of rigs have added this feature and this, too, is usually compatible with the Raspberry Pi 3.

Regardless of which model you choose, I recommend an interface with a built-in sound device. Otherwise, you'll have to deal with providing sound inputs and outputs to the Pi. The Raspberry Pi has sound capability, but it is limited to a sound *output*; you need a sound *input* as well. One solution is to purchase an inexpensive USB sound device (typically selling for less than $20 from vendors such as Amazon) and use it to transfer audio to and from your Pi.

- **An SD memory card**. Some kits include a memory card. If not, I recommend an 8 GB card or larger — rated at a speed of 10. Some Raspberry Pi kits that include SD cards also provide SD card readers that you can plug into a USB port on your primary computer. These are important because, as you'll see in a moment, you'll have to initially download an *image file* for your Pi's operating system using your primary computer, and then write the image file to the card. If your Raspberry Pi package didn't include a card reader, you can purchase one online for $10 or less.

Setting up Your Raspberry Pi

Step One: Download an image of the latest *Raspbian Jessie with Pixel* operating system. This operating system is a version of *Linux*, and it is fully compatible with the Raspberry Pi 3. You will find it here: **https://www.raspberrypi.org/downloads/raspbian/**. Choose the ZIP file and download it to your primary computer hard drive.

Step Two: Write the image file to the SD card. Here is where you'll need your SD card reader. You'll also need specialized image-writing software to do this, but the software is free and easy to use. You'll find sources for image-writing software at **https://www.raspberrypi.org/documentation/installation/installing-images/**.

Step Three: Assemble your Raspberry Pi 3 and attach all the cables — *except the power supply*. We don't want to apply power quite yet. If your HF transceiver has a built-in interface, you will need to turn on the transceiver. If you are using an external interface unit, you will need to power it up.

Step Four: Plug the SD card with the *Raspbian* operating system into the Raspberry Pi 3. Now plug in the 5 V power cable as well. *Raspbian Jessie* will boot up on your monitor.

Getting Started with *WSJT-X*

Step One: Configure the waterfall window. With the mouse, reposition the two *WSJT-X* windows so that they don't overlap. In the waterfall window, check/set the following parameters:

Bins/Pixel = 4
Start = 100
N avg = 5
Spec = 40
Flatten (check)
Cumulative (check)
Palette — your choice on this one. I use the YL2KF palette.

Step Two: Enter your call sign and grid square. Click on the main *WSJT-X* window and press the F2 key. In the GENERAL tab, type in your call sign and your Maidenhead coordinates. Generally, you need only the first two letters and the next two numbers. As an example, the Maidenhead coordinate for my station is EM73. If you don't know your grid square, Dave Levine, K2DSL, has created a handy online calculator at **www.levinecentral.com/ham/grid_square.php**.

Step Three: Set up the control for your HF transceiver. Of course, this presumes that you have a transceiver interface that provides transceiver control (often called "CAT" control), or that you have a direct connection to your radio.

Click on the RADIO tab.
Rig — select your transceiver model from the list.
Poll interval — set to 1 second.
CAT control area

Serial port = /dev/ttyUSB0
Baud rate — If you don't know the baud rate, start with 9600. Use the TEST CAT button to verify. Keep raising the rate until the TEST CAT button indicates a failure. Then, back reduce the rate to the next lowest speed.

PTT method — set to CAT
Mode — set to Data/Pkt
AGC — set to Fast

Step Four: Set up the audio devices for *WSJT-X*. Click on the AUDIO tab. For the Input and Output pull-down menus, select "sysdefault:CARD=CODEC"

Step Five: Set up the Reporting menu. This allows *WSJT-X* to upload your reception reports automatically to the internet. Click and checkmark the following:

- Prompt me to log QSO
- dB reports to comments
- Clear DX call and grid after logging

And finally, click the OK button.

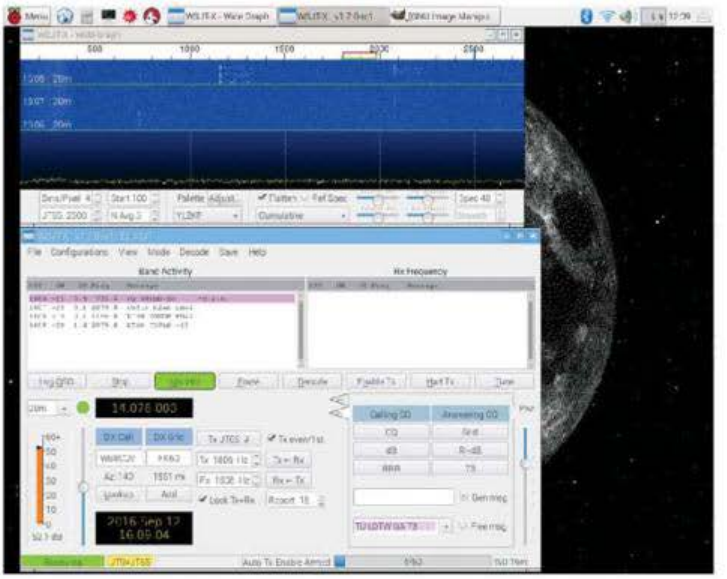
On the main window of *WSJT-X*, click and checkmark LOCK TX=RX, unless you are doing split-frequency operating. The other item to checkmark is KEEP A COPY in the QSO logging window. By making a checkmark on this item, *WSJT-X* will remember every logged QSO. This way, if you see their call sign appear in the receive window, they will be colored green, indicating that you have already logged a QSO with them before.

Now you're ready to go. Before you attempt your first contact, check your transceiver settings to make sure they are compatible. For example, some transceivers have a built-in digital mode setting that you may have to select. Also, be sure to disable DSP noise reduction, speech processing, and noise blanking.

If you are new to the modes offered by *WSJT-X*, I strongly recommend that you do some reading before taking to the airwaves. A good source of information is the ARRL book *Work the World With JT65 and JT9* by Steve Ford, WB8IMY, available from your dealer or online at **www.arrl.org/shop**.

Step Five: **Configure localization**. Click the menu button in the upper left corner of the screen, and then select Preferences, Raspberry Pi Configuration, and then click the Localization tab. In this window, you'll simply set up the Pi to conform to your geographic location. When you finish, click OK. The Pi will ask, "Would you like to reboot now?" Click YES.

Step Six: Set up the default audio device. In this step, you'll tell the Pi which audio device you are using — either a sound device plugged into one of the Raspberry Pi USB ports, an external interface with a built-in sound device, or a USB cable directly attached to your transceiver. Regardless of which device it may be, it will usually be labeled "USB Audio



WSJT-X running on the author's Raspberry Pi 3 under the *Raspbian Jessie with Pixel* operating system.

Time and the Raspberry Pi 3

Modes such as JT65, JT9, and WSPR all have at least one thing in common: they all require accurate time. The transmit and receive sequences are timed to the second, which means your Raspberry Pi's clock must also be accurate to the second.

If your Raspberry Pi 3 is connected to the internet, the default time server used by *Raspbian Linux* works quite well. But if you don't have an internet connection, you will need an alternative. One is a so-called real-time clock module that plugs into the Raspberry Pi board and is available on Amazon for less than $10. Another, and more accurate, alternative is to take a second Raspberry Pi (Version 2) and convert it into a GPS-based time server. You can find the details I've compiled online at **www.satsignal.eu/ntp/Raspberry_Time%20-%20Broadband%20Ham%20Net.pdf**.

Codec," or something similar.

Again, go to the menu, and then select PREFERENCES, followed by AUDIO DEVICE SETTINGS. From the pull-down menu at the top of the SETTINGS window, select the USB Audio Codec device. Click on the SELECT CONTROLS button at the bottom. In the Controls panel, click and checkmark the PCM listed and click on CLOSE. The tab will now be labeled as PLAYBACK, and you will see two sliders appear. Click on the MAKE DEFAULT button at the bottom and then click on OK.

Step Seven: Connect the Raspberry Pi to the internet. If you are using an ethernet cable for your internet connection, simply plug in the ethernet cable. I am presuming that the other end of the ethernet cable is plugged into a device which accesses the internet (like a router or cable modem).

For a wireless setup, look at the right side of the top bar. You will see a Bluetooth icon and, to the right of that icon, the network icon. Left-click on the network icon and select your wireless router's SSID from the list. If a passphrase is needed (encrypted connection), you will be asked for the "Pre Shared Key." Type in the password and click OK.

Now that your Raspberry Pi 3 is up and running, a word of caution: do not use the version of *WSJT-X* that comes prepackaged with *Raspbian Linux*. That version of *WSJT-X* is flawed and does not work well. Instead, download a version compiled by Greg Beam, KI7MT, which works flawlessly on the Raspberry Pi 3 (**http://wsprnet.org/drupal/node/5567**). Use the Debian Installation instructions that appear on the website.

Thomas Kocourek, N4FWD, learned electronics in the military during the Vietnam War era. He attended several technical schools after leaving military duty and is now retired from a career as an MRI engineer. You can contact him at **n4fwd@yahoo.com**.

For updates to this article, see the *QST* Feedback page at www.arrl.org/feedback.

Coming up in the July/August 2017 and future Issues…

A new column appears in *QEX*. We invite you to share a brief **Technical Note** that is several hundred words long, with a figure or two. Make your technical observation, or expand briefly on another author's work and add to the Amateur Radio *institutional memory*.

Our *QEX* authors touch upon many aspects of Amateur Radio — from filters and circuits, to antenna analysis, and a new communications protocol. These topics are in the queue.

- Riccardo Gionetti, I0FDH, describes an automatic tracking filter for a DDS generator.
- Euclides Lourenço Chuma, PY2EAJ, describes a modern RF power meter with accuracy that rivals the best commercial RF power meters.
- Robert J. Zavrel, W7SX, explains field-strength and power combining in various configurations of dipoles and dipole arrays.
- Steven J. Franke, K9AN, and Joseph H. Taylor, K1JT, describe the modulation, message structure, channel coding, and special operational features of the new meteor-scatter mode implemented in WSJT-X.
- Scott Roleson, KC7CJ, describes a high-dynamic range broadband amplifier that enhances the usability of an SDR or any HF receiver.
- Charles Preston, K7TAA, uses WSPR (Weak Signal Propagation Reporter) software to facilitate reliable and accurate comparison of two HF transmitting antennas.

QEX is edited by Kazimierz "Kai" Siwiak, KE4PT, (**ksiwiak@arrl.org**) and is published bimonthly. *QEX* is a forum for the free exchange of ideas among communications experimenters. The content is driven by you, the reader and prospective author. The subscription rate (6 issues per year in the United States is $29. First Class delivery in the US is available at an annual rate of $40. For international subscribers, including those in Canada and Mexico, *QEX* can be delivered by airmail for $35 annually. Subscribe today at **www.arrl.org/qex**.

Would you like to write for *QEX*? We pay $50 per published page. Get more information and an Author Guide at **www.arrl.org/qex-author-guide**. If you prefer postal mail, send a business-size self-addressed, stamped (US postage) envelope to: *QEX* Author Guide, c/o Maty Weinberg, ARRL, 225 Main St, Newington, CT 06111.